IN THE CLAIMS

1. through 15. (Canceled)

16. (Currently Amended) A computer-implemented method for measuring a complexity of nested object state transition diagrams that are represented as data structures in the computer, the method comprising ~~the steps of~~:

a) using a computer, determining a plurality of graphs of object state transitions at K levels l_k, wherein:

(A) $0<k<K$; ~~wherein~~

(B) one or more of said graphs at level l_k+1 are expansions of one or more of said graphs at level l_k; and

(C) said graphs comprise a plurality of nodes to represent a corresponding plurality of states of use-cases and a plurality of edges to represent a corresponding plurality of transitions between the states; and

b) using the computer, determining ~~the~~ a complexity for said plurality of graphs.

17. (Currently Amended) The computer-implemented method ~~as in~~ of claim 16, wherein:

the state of an ~~ob~~ object at level k is a super state of an object state at level k+1, and

~~it~~ the state of an object at level k is a sub state of an object state at level k-1, and

the method further includes ~~the step of~~ identifying the state transion relationship between super state and sub state as ~~in~~ one of a group consisting of "in," "out," and "inout". ~~the three cases as "in" / "out" / "inout".~~

18. (Currently Amended) The computer-implemented method ~~as in~~ of claim 17, further comprising: ~~wherein said determining the number of paths comprises the steps of~~

selecting said level $l_k = l_K$;

expanding a graph L from said selected level l_k with ~~said~~ at least one graph from said level l_{K+1};

determining ~~the~~ paths in said expanded graph; and

determining ~~the~~ a number of conditions in said transition paths.

19. (Currently Amended) The computer-implemented method ~~as in~~ of claim 18 further comprising: ~~the step of~~

removing ~~one or more~~ an unnecessary path [s] from said at least one graph.

20. (Currently Amended) The computer-implemented method ~~as in~~ of claim 19 wherein said one or more unnecessary paths comprise one or more numbers of ~~the~~ a group consisting of:

(exception → fallout),

(exception → exception),

(one-repetition loop →fallout),

(one repetition loop → cancelled), and

(one repetition loop → exception).

21. (Currently Amended) A computer-implemented method ~~as in~~ of claim 16 to measure a nested object state transition complexity between two super-states by recursively applying Equation 6:

$$STPC_{k,p,q} = \sum_{i=1}^{m_k}\left(\prod_{j=1}^{n_{k,i}}(C_{k,i,j} + (STPC_{k+1,i,j} - sub_{k,i,j})) + mul_{k,i}(N)\right) \quad \text{Equation 6}$$

wherein:

m_k = ~~the~~ a number of transition paths of level-k object (k>1) between two super-states or ~~the~~ a number of transition paths of level-1 object;

$n_{k,i}$= ~~the~~ a number of states along path i for level-k object; [.]

$C_{k,i,j}$ is ~~the~~ a number of conditions to bring level-k object state from Sk,i,j-1 to Sk,i,j ;

$STPC_{k+1,i,j}$ = Substate Transition Complexity between state Sk,i,j-1 and Sk,i,j;

$sub_{k,i,j}$=0 if $STPC_{k+1,i,j}$=0 and $sub_{k,i,j}$=1 if ($STPC_{k+1,i,j}\neq 0$) ;

~~If the~~ if a multiplicity between level-k ~~objet~~ object and $level_{-(k+1)}$ is 1:N, then $mul_{k,i}(N)$=0 [,] if (N=1) and $mul_{k,i}(N)$=r [,] if (N>1); and

~~Where~~ r is ~~the~~ a number of times that $level_{-k+1}$ objects transit [s] back to $level_{-k}$ in path i.

22. (Canceled)